REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE

2. REPORT TYPE
Paper

3. DATES COVERED

4. TITLE AND SUBTITLE

Using HLA to Transmit Real-Time Sensor Imagery

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Mark Silbert; William Schibler; Gordon Curran

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Air Warfare Center Aircraft Division
22347 Cedar Point Road, Unit #6
Patuxent River, Maryland 20670-1161

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

An initiative, called Modular Avionics Integration Network/Modular Avionics Integration Laboratory (MAIN/MAIL), was started in FY99 to create a coordinated approach among NAWCAD laboratories to support more efficient use of both external and internal capabilities and facilities through interoperable network connectivity. This three-year effort will ensure that the Navy makes maximum use of the laboratory resources available, and through networking, provides a capability for multiple center participation in shared program developments.

MAIN/MAIL uses the Defense Modeling and Simulation Office (DMSO) High Level Architecture (HLA) to facilitate this interoperability and reusability. To demonstrate the capability, three demonstrations have been planned each showing progressively more sophisticated data sharing. These demonstrations have been labeled Demo A, B, and C.

A successful demo system for Demo A was completed in FY00. The purpose of Demo A was to develop a prototype, real-time system that would share real-time, real sensor imagery data among three laboratories physically located within a building at NAWCAD Patuxent River, Maryland. These labs were connected using a fiber optic, local area network. The sharing of the imagery data over the network was accomplished through software that uses HLA. The three laboratories involved in this data sharing were the AVX Sensor Lab, the IFV (Image Fusion and Visualization) Lab, and the CTL (Crew Technology Lab). The AVX is an Electro-Optical (EO) sensor that provided the real-time imagery. Imagery from the AVX Lab was broadcast via HLA to the other two labs. Upon receiving the imagery data in the IFV Lab, the real-time images were provided as input to an algorithm that "identifies" stationary objects in the image and creates annotations for those objects. The annotations for each image were then sent via HLA to the CTL. Once the CTL received both the imagery data from the AVX Lab and the annotations from the IFV Lab, it would display the annotated image. To complete the loop, the operator viewing the annotated image in the CTL was able to control the position of the AVX sensor via his keyboard or joystick. This sensor control was then sent back to the AVX lab, via HLA to move the sensor to the new position.

The key issues of this effort were how fast a frame rate was achievable and how to configure the HLA/system to achieve this frame rate. This led to a study to determine how to configure the HLA to achieve the best frame rate. Several alternatives were considered including using HLA "best-effort" and "reliable" protocols, regulating/nonregulating modes, and video compression. The study made it clear that these factors could dramatically impact the frame rate and/or reliability of the system. Depending on the configuration, we found that it was possible to achieve sustained frame rates as high as 10 frames per second. The follow-on demonstrations B and C, will use HLA to pass more control information and send this data over wider local area networks as well as over wide area networks.

15. SUBJECT TERMS
HLA, RTI, sharing assets, video, sensors

16. SECURITY CLASSIFICATION OF:

a. REPORT

b. ABSTRACT

c. THIS PAGE

17. LIMITATION OF ABSTRACT

18. NUMBER OF PAGES
5

19a. NAME OF RESPONSIBLE PERSON
Mark Silbert, William Schibler, Gordon Curran

19b. TELEPHONE NUMBER (include area code)
(301) 342-9111 / 0069 / 2089

Using HLA to Transmit Real-Time Sensor Imagery

Mark Silbert, William Schibler, Gordon Curran
Naval Air Warfare Center, Aircraft Division
Patuxent River, MD 20670

Keywords HLA, RTI, sharing assets, video, sensors

Abstract
An initiative, called Modular Avionics Integration Network/Modular Avionics Integration Laboratory (MAIN/MAIL), was started in FY99 to create a coordinated approach among Naval Air Warfare Center (NAWC) laboratories to support more efficient use of both external and internal capabilities and facilities through interoperable network connectivity. This three-year effort will ensure that the Navy makes maximum use of the laboratory resources available, and through networking, provides a capability for multiple center participation in shared program developments.

MAIN/MAIL uses the Defense Modeling and Simulation Office (DMSO) High Level Architecture (HLA) to facilitate this interoperability and reusability. To demonstrate the capability, three demonstrations have been planned each showing progressively more sophisticated data sharing. These demonstrations have been labeled Demo A, B, and C.

A successful demonstration system for Demo A was completed in FY00. The purpose of Demo A was to develop a prototype, real-time system that would share real-time, real sensor imagery data among three laboratories physically located within a building at NAWC-AD Patuxent River, Maryland. These labs were connected using a fiber optic, local area network. The sharing of the imagery data over the network was accomplished through software that uses HLA. The three laboratories involved in this data sharing were the AVX Sensor Lab, the IFV (Image Fusion and Visualization) Lab, and the CTL (Crew Technology Lab). The AVX is an Electro-Optical (EO) sensor that provided the real-time imagery. Imagery from the AVX Lab was broadcast via HLA to the other two labs. Upon receiving the imagery data in the IFV Lab, the real-time images were provided as input to an algorithm that "identifies" stationary objects in the image and creates annotations for those objects. The annotations for each image were then sent via HLA to the CTL. Once the CTL received both the imagery data from the AVX Lab and the annotations from the IFV Lab, it would display the annotated image. To complete the loop, the operator viewing the annotated image in the CTL was able to control the position of the AVX sensor via his keyboard or joystick. This sensor control was then sent back to the AVX lab, via HLA to move the sensor to the new position.

The key issues of this effort were how fast a frame rate was achievable and how to configure the HLA/system to achieve this frame rate. This led to a study to determine how to configure the HLA to achieve the best frame rate. Several alternatives were considered including using HLA "best-effort" and "reliable" protocols, regulating/non-regulating modes, and video compression. The study made it clear that these factors could dramatically impact the frame rate and/or reliability of the system. Depending on the configuration, we found that it was possible to achieve sustained frame rates as high as 10 frames per second. The follow-on demonstrations B and C, will use HLA to pass more control information and send this data over wider local area networks as well as over wide area networks.

INTRODUCTION

An initiative, called Modular Avionics Integration Network/Modular Avionics Integration Laboratory (MAIN/MAIL), was started in FY99 to create a coordinated approach among Naval Air Warfare Center (NAWC) laboratories to support more efficient use of both external and internal capabilities and facilities through interoperable network connectivity. This three-year effort will ensure that the Navy makes maximum use of the laboratory resources available, and through networking, provides a capability for multiple center participation in shared program developments.

The need to share laboratory assets is very real. In today's fiscally prudent environment, more productivity is expected from the labs, yet funding dollars have been reduced. A key way to overcome the insufficient funding, is to have the labs share their assets so that each lab can create higher-fidelity development environments and be more productive. In order to share lab resources effectively, we need to use an open and standard set of hardware and software. With this is mind, we chose to use the DoD standard network software protocol of High Level Architecture Real-Time Initiative (HLA-RTI) since is becoming the de facto standard for the DoD modeling and simulation community. In order to share

the lab assets effectively, control of the resource must be immediate and responsive. Besides the ability to share resources, MAIN/MAIL will yield other benefits. By employing the actual resources used, system integration can be accelerated and started sooner than it has been able to do in the past. This will reduce development cycle time and costs. Furthermore, by integrating science and technology, research and development with the test and evaluation, we provide a means for testing to be done sooner in the pipeline.

DEFINING THE STARTING POINT

We realized from the onset that in order for the laboratories to share resources effectively, we needed to show that the selected set of standards could support immediate and responsive control and feedback of any resource. Since we had chose HLA-RTI to be the network data interchange protocol, we needed to show that HLA-RTI supports our requirements. We decided to choose a real Navy sensor that produces a video output. The reasoning was that video has high demands in terms of message size and message rates. Namely, in order to maintain a smooth video stream, the HLA must be able to support sending large video images at a high repetition rate. Simply stated, if we show we can share a video stream, we've helped proved that HLA should be able to support nearly any kind of laboratory asset. Furthermore, by sharing video, there are obvious quantifiable measures we can make on performance such as frame size and frame rate. In addition, by using compression we can see how the frame rate improves as the frame size (i.e., message size) decreases. This reasoning defined our first demonstration, Demo A.

THE STARTING POINT - DEMO A

The architecture for Demo A is shown in figure 1. As the figure shows, Demo A primarily consists of an electro-optical (EO) sensor called the AVX and three personal computers (PCs) networked together on a local area network (LAN) within a single building. Both the AVX and the CTL PCs were each a Pentium II 450, the IFV PC was a Pentium II 400. The HLA RTI was used as the data interchange protocol over the LAN. Each of the three computers were stationed in three labs; the AVX lab, the IFV lab, and the CTL.

The AVX computer was physically connected to the AVX sensor. To connect the sensor to the computer we needed to install two special expansion cards into the AVX computer, and frame grabber (FG) and a digital-to-synchro (D/S) card. The FG card takes in the video output signal from the sensor and digitizes it into a video bitmap. We used the Matrox Meteor II frame grabber card. The digitized video was then shared with the other two computers using RTI through a multicast interaction. The D/S card provides control of the sensor from the computer. The D/S converts a digital number into a synchro command that is then given to the sensor's servomotors to move the sensor. The D/S card was an ISA-based interface card purchased from DDC. As part of Demo A, software was written to communicate with both the FG and the D/S cards.

The IFV computer was running special image annotation software that was developed for the MAIN/MAIL project. The annotation software takes a digital video frame as input, locates (stationary) objects in the frame and then outputs a list of annotations for all the objects found. As the IFV computer receives each digital video frame, it processes the frame using this annotation software and then forwards the annotations to the CTL computer, via the RTI.

The CTL computer receives the digital video image from the AVX computer and the list of annotations for that frame from the IFV. The CTL then displays the image, frame-by-frame, with the annotations overlaid over the image. To compute the control loop, the CTL allows the user to control the AVX sensor via a joystick or the keyboard. These control commands are sent back to the AVX computer via the RTI. Once the AVX receives the command, it moves the AVX sensor by sending the control request to the D/S controller card.

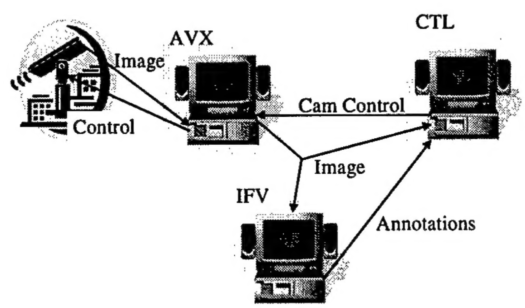

Figure 1. Demo A architecture.

A block diagram of Demo A is shown in figure 2. All data interchanges between the computers were accomplished using RTI interactions. Interactions were used since there was no need for any of the time-regulating tools provided by the RTI. Each of the three computers performed a simple processing loop for each video frame. Figure 3 shows the simplified processing loop that was performed on each computer. During each loop, each process checked for an interaction by making a call to the RTI *Tick* function.

One key ingredient that has been omitted in figure 3 is how we kept the video frame synchronized with the annotations. Recall that the AVX multicasts the video frame to both the

IFV and the CTL simultaneously. Once the image is received at the IFV, further processing is necessary to compute the annotations for the image, which must be subsequently sent on to the CTL as well. Without any synchronization, the annotations would end up being displayed at the CTL on a later image. To ensure that the annotation and image were displayed together, a synchronization scheme was implemented as follows. First, before each image is sent from the AVX, it is assigned a frame ID number. This frame ID number, which is simply a counter that is used to identify the frame, was sent along with each video frame. When the annotations are created at the IFV node, it sends them along with the received frame ID. When the CTL receives the frame image, it then waits for the annotations with that same frame ID to arrive as well. It then displays the annotations on the received image. As a final step, after the CTL displays the image, it needs to alert the AVX that it is now ready to receive the next frame. This alert or acknowledgement message is implemented as an RTI interaction as well.

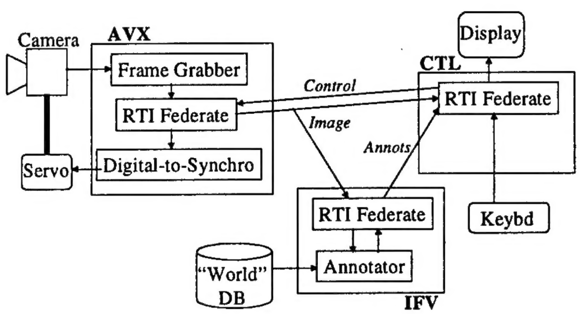

Figure 2. Demo A block diagram.

```
AVX                               CTL
LOOP                              LOOP
  Get camera position;              Get frame;
  Move camera;                      Display frame;
  Grab frame;                       Get annots posits;
  Send frame;                       Display annots on image;
  Tick the RTI;                     Read camera pos from user;
ENDLOOP                             Send camera position;
                IFV                 Tick the RTI;
                LOOP              ENDLOOP
                  Get frame;
                  Determine objects in frame;
                  Determine posits for annots;
                  Send annot posits;
                  Tick the RTI;
                ENDLOOP
```

Figure 3. Demo A processing loop for each computer.

Although the description of processing for each computer given is functionally correct, there are some other details that need to be accounted for. Namely, we have noticed that some of the RTI interactions go unnoticed by the receiving process. These unnoticed messages occur particularly when the RTI is running in best-effort mode. When an unnoticed interaction occurs, it is experienced as a missed or dropped message. Logic needs to be added to the processing loops to time-out when an anticipated interaction doesn't occur.

DEMO A RESULTS

Once we got the system working, we studied how fast we could push frames through by computing the frame rate. The three computers were connected using 100BaseT network cards. The 100BaseT cards and network were both rated at 100Mbits/sec. The (uncompressed) images that were sent were digitized at a resolution of 640 x 480 pixels, with 8 bits/pixel yielding 300KB per frame. Dividing the bandwidth by the frame size (and subtracting out the network overhead) we arrive at a theoretic upper limit of approximately 30 frames/sec (FPS). However, using the RTI "reliable" protocol, the frame rate was only about 3.2 FPS; nearly an order of magnitude lower than the theoretical limit. A study was conducted to see why the frame rate was so low. Table 1 shows results from some of our studies. Note that all the frame rates are well below our 30 FPS limit. The study yielded several key factors attributing to the low rate. These key factors include the speed of the computer, reliable versus best-effort protocols for the interactions, and the time to grab and digitize the video image. Image size had some impact on the results as well but not as much as expected.

Using AVX EO Sensor

	Raw Video (300KB)	JPEG (≈13KB)
Best-Effort	6.4	5.7
Reliable	3.2	3.5

Using Image Files

	BMP Image (300KB)	JPEG (≈13KB)
Best-Effort	9.1	9.6
Reliable	3.2	9.6

Table 1. Demo A study results.

Computer Speed

Clearly the faster the computer can process its loop, the higher the frame rate will be. Running the Windows System Monitor on the computers showed that the machines were running in the sustained 80-90% utilization levels. Basically our Pentium II 450 computers were being "maxed out".

Unfortunately, we did not have any faster computers available at the time to see how much higher of a frame rate would be achieved using faster computers.

Reliable vs Best-Effort

Initially, all data passing between the RTI nodes was accomplished using RTI's reliable protocol. With this protocol, the RTI makes the extra effort to ensure the interactions are successfully completed. Using this protocol yielded a fairly robust simulation but at a fairly high cost – it dramatically reduced the frame rate performance. We decided to try the best-effort protocol. When we switched to the best-effort protocol, the frame rate doubled from 3.2 FPS to 6.4 FPS. However, our improved performance also came at a price. Letting the system run for awhile, we would notice that the system would typically hang at seemingly random intervals. Apparently, the system was waiting for a message that never arrived. We modified our simulation by putting time-outs on the arriving messages. While this cleared up some of the problems, the best-effort protocol never always worked reliable. The best-effort protocol earned the nickname as the "unreliable" protocol.

Memory Buffer vs Video Frame

Although we were always grabbing the last available digital video buffer, we wondered if there still was any lag being introduced from the grab itself. To test for this effect, we saved a video buffer to an array in memory. Then, instead of grabbing the video frame, we repeatedly sent the video buffer that was saved in memory. In reliable mode, the frame rate went nearly unchanged implying that there was no loss in time from grabbing the video buffer. This is what we hoped would happen. Our expectation, however, was short lived. We then performed the same test using best-effort protocol. We expected that the frame rate should parallel the best-effort protocol with video and yield the frame rate of 6.4 FPS. Instead, a dramatic increase in frame rate occurred. The frame rate jumped to 9.1 FPS. Furthermore, the system acted more stable than it did when we sent the video buffers across. We still do not have a good explanation for these results.

Image Size

We also tested to see if reducing the frame size would help increase the frame rate performance since smaller messages should process faster. To verify this, we took (compressed video) JPEG files, stored them in an array in memory and then sent them as the video frames. These JPEG buffers were approximately 13KB each, substantially smaller than the raw video frames of 300KB. In reliable mode, there was a dramatic increase in frame rate from 3.2 FPS to 9.6 FPS. Switching to best-effort mode yielded the same frame rate of 9.6 FPS. We decided to purchase a hardware JPEG encoder. We purchased the Matrox Meteor II JPEG encoder module which plugged into the Matrox Meteor II frame grabber board we were already using. We then tested the frame rate sending the compressed video. The Meteor II compression board provides program-settable levels of compression via a compression quantitization parameter. We selected quantitizations that produced frame sizes of approximately 9KB and 13KB. The results were much less than expected. Using the reliable protocol, there was some increase over sending the raw video but is was not dramatic. In best-effort mode, the frame rate actually went down. What accounted for this less than stellar performance? It was determined that there were two problems. The first problem was that the encoder required about 30 msecs to compress each video frame, taking up valuable frame rate time. A quick calculation shows that the network has enough bandwidth to send an entire, uncompressed video frame in that 30 msec interval (i.e., 30 msecs x 100 Mbit/sec = 300KB). So the time saved in the transfer of a smaller file didn't outweigh the time needed to compress it. The second problem was that there seemed to be a resource that was being shared between the JPEG decoder and the RTI itself. This was evidenced by the fact that initially the system would hang until we entered Sleep(0)'s in the process loops. Namely, we needed to force the processes to relinquish control of whatever resource they were sharing so that the system would run normally.

LESSONS LEARNED – ON TO DEMO B

Although Demo A raised many more questions about performance than we expected, it was clear that sending video can be problematic in RTI. There needs to be time to tweak and experiment with these critical factors to get a simulation that is fast, responsive and robust. A key lesson to be learned was to try to find an alternative approach if large messages needs to be sent at high update rates.

In Demo B, we take heed of our own advice. The purpose of Demo B is to extend and enhance the capability developed in Demo A. The architecture for Demo B is given in figure 4. In this demo, there are two buildings, a radar building and a lab building housing a P-3 airplane mock-up facility. These buildings are separated about 4 miles apart but are connected by both a fiber optic LAN and a fiber optic link. The radar building has a APS radar system very similar to the radar system on a P-3. It produces a video signal of radar detections. The objective in Demo B is to be able to use and control the radar from the P-3 facility. Currently, only the radar station within the radar building can control the radar. The P-3 lab would like to extend the capability of their lab by being able to use and control the radar located 4 miles away. In order for the P-3 lab to utilize the radar effectively, control and response must be immediate, like it is on the P-3

airplane itself. With the results from Demo A, it was clear that RTI could not support the high data rates needed. To off-load the RTI, we decided to channel the video output over a separate piece of fiber leaving RTI to only handle the control of the radar. In addition, it was clear that high-end computers should be used to handle the high-speed interaction. For Demo B, we are using a Pentium III 933 in the radar building and a Pentium III 866 in the P-3 lab facility. Using ADC equipment, we have already successfully shown that we can send the radar video immediately and continuously. We are scheduled to demonstrate the radar control capability via the RTI by mid April 2001.

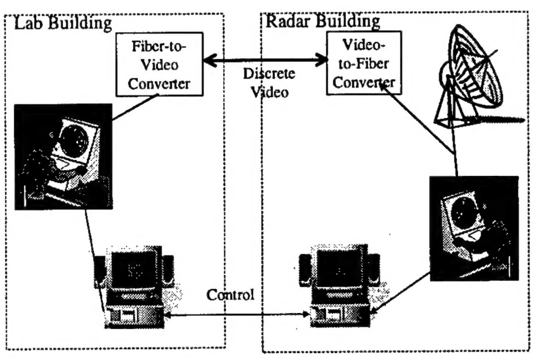

Figure 4. Demo B architecture.

MULTI-ASSET SHARING OVER A WAN - DEMO C

We are currently coordinating with NAWC-TSD in Orlando FL and NAWC-??? in Lakehurst, NJ to develop our final demonstration, Demo C. The architecture for this demo is given in figure 5.

The intent of this demo is to show that each of the three Navy bases can control and use an asset owned by a particular base. Each base is currently tasked to identify an asset that they own and would be willing to share with the other bases. The detailed control (i.e., inputs) and outputs will then be identified for each asset. The computers will exchange control and output data via the RTI over a WAN. Each base will multicast the control and output of the asset so each base will have an up-to-date view of all three assets. This demo will highlight the power of sharing resources by allowing each base to control and use an asset physically located at another base. A demonstration of this capability is scheduled for the end of FY-2001.

CONCLUSION

In order for the HLA RTI to be successful as the data interchange network protocol, it must be efficient and robust enough to support large messages and high update rates. Our experience has shown that configuring and "tuning" the RTI to accomplish these demands can be troublesome. In addition, some configurations lead to surprising results. Hopefully, as the computers and networks become faster and the RTI matures, it will be easier to configure these systems.

To test how faster computers will improve performance, we plan to re-host Demo A on the new computers purchased for Demo B. Barring any surprises, we hope to show that by using the faster computers, the frame rate will increase as expected.

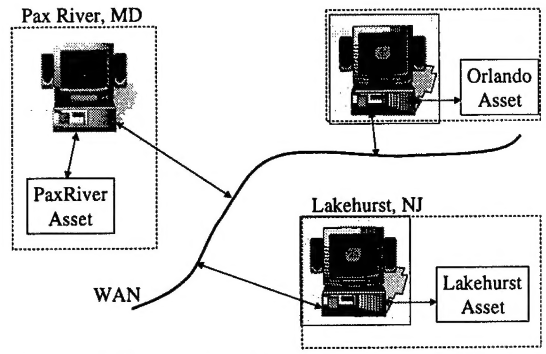

Figure 5. Demo C architecture.

Biography

Mr. Mark Silbert is a data/information fusion engineer working at the NAWCAD for 22 years. His background is in the area of data fusion and artificial intelligence. Mr. Silbert has earned a B.S. degree in computer science from Drexel University and a M.S. degree in computer science from Rutgers University. Over the 22 years, Mr. Silbert has worked on many data fusion and decision-aiding systems. He was also an adjunct professor for two colleges teaching artificial intelligence and statistics.

Mr. William Schibler...

Mr. Gordon Curran has been an Analog/Digital Signal Processing Engineer in the Acoustic Signal Processing Branch at NAWC-AD Patuxent River for the past five years. Prior to that, he performed 35 years of Navy-orientated engineering services as a contractor for RCA. Mr. Curran did his undergraduate work at the Moore School of Engineering (University of Pennsylvania) and received an M.S.E.E degree (Acoustics) from Pennsylvania State University. His contract efforts include analysis of submarine sonar systems, Naval standard signal processing software support for the AN/UYS-2, and worked on a weather satellite for NASA.